	Type	L #	Hits	Search Text	DBs	Time Stamp	Comments
1	BRS	L1	0	addressing WITH preindirecting	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/09/14 14:38	
2	BRS	L2	0	addressing SAME preindirecting	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/09/14 14:39	
3	BRS	L3	0	addressing SAME preindirect$5	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/09/14 14:39	
4	BRS	L4	0	preindirect$5	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/09/14 14:39	
5	BRS	L5	2857	addressing SAME indirect$5	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/09/14 14:39	
6	BRS	L6	1310	5 and (register WITH indirect$5)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/09/14 14:40	
7	BRS	L7	320	6 and program adj memory	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/09/14 14:40	

	Type	L #	Hits	Search Text	DBs	Time Stamp	Comments
8	BRS	L8	4	7 and variable adj memory	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/09/14 14:43	
9	BRS	L9	0	7 and preindex$5	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/09/14 14:43	
10	BRS	L11	0	7 and pre-indirect$5	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/09/14 14:43	
11	BRS	L10	9	7 and pre-index$5	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/09/14 14:43	

Terms used
address and **program** and **variable** and **table** and **memory** and **branching** and **bus** and **data**

Found **87,360** of **142,346**

Sort results by: relevance

Display results: expanded form

Save results to a Binder

Search Tips

Open results in a new window

Try an Advanced Search

Try this search in The ACM Guide

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next

Best 200 shown

Relevance scale

1 Cache Memories

Alan Jay Smith

September 1982 **ACM Computing Surveys (CSUR)**, Volume 14 Issue 3

Full text available: pdf(4.61 MB) Additional Information: full citation, references, citings, index terms

2 Data and memory optimization techniques for embedded systems

P. R. Panda, F. Catthoor, N. D. Dutt, K. Danckaert, E. Brockmeyer, C. Kulkarni, A. Vandercappelle, P. G. Kjeldsberg

April 2001 **ACM Transactions on Design Automation of Electronic Systems (TODAES)**, Volume 6 Issue 2

Full text available: pdf(339.91 KB) Additional Information: full citation, abstract, references, citings, index terms

We present a survey of the state-of-the-art techniques used in performing data and memory-related optimizations in embedded systems. The optimizations are targeted directly or indirectly at the memory subsystem, and impact one or more out of three important cost metrics: area, performance, and power dissipation of the resulting implementation. We first examine architecture-independent optimizations in the form of code transoformations. We next cover a broad spectrum of optimizati ...

Keywords: DRAM, SRAM, address generation, allocation, architecture exploration, code transformation, data cache, data optimization, high-level synthesis, memory architecture customization, memory power dissipation, register file, size estimation, survey

3 Compiling nested data-parallel programs for shared-memory multiprocessors

Siddhartha Chatterjee

July 1993 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 15 Issue 3

Full text available: pdf(4.17 MB) Additional Information: full citation, references, citings, index terms, review

Keywords: compilers, data parallelism, shared-memory multiprocessors

4 Performance evaluation and improvement of a dynamically microprogrammable computer with low-level parallelism

Shinji Tomita, Kiyoshi Shibayama, Toshiaki Kitamura, Hiroshi Hagiwara

November 1980 **Proceedings of the 13th annual workshop on Microprogramming**

Full text available: pdf(1.21 MB) Additional Information: full citation, abstract, references, citings, index terms

A new microprogrammable computer with low-level parallelism was built and has been utilized as a research vehicle for solving different classes of research-oriented applications such as real-time processings on static/dynamic images, pictures and signals, and emulations of both existing and virtual machines including high (intermediate) level language machines. The design goal of a research-oriented computer, QA-1, was to achieve a high degree of processing power and system flexi ...

5 The Clipper processor: instruction set architecture and implementation

W. Hollingsworth, H. Sachs, A. J. Smith

February 1989 **Communications of the ACM**, Volume 32 Issue 2

Full text available: pdf(4.67 MB) Additional Information: full citation, abstract, references, citings, index terms, review

Intergraph's CLIPPER microprocessor is a high performance, three chip module that implements a new instruction set architecture designed for convenient programmability, broad functionality, and easy future expansion.

6 Tango: a hardware-based data prefetching technique for superscalar processors

Shlomit S. Pinter, Adi Yoaz

December 1996 **Proceedings of the 29th annual ACM/IEEE international symposium on Microarchitecture**

Full text available: pdf(1.46 MB) Publisher Site Additional Information: full citation, abstract, references, citings, index terms

We present a new hardware-based data prefetching mechanism for enhancing instruction level parallelism and improving the performance of superscalar processors. The emphasis in our scheme is on the effective utilization of slack time and hardware resources not used for the main computation. The scheme suggests a new hardware construct, the program progress graph (PPG), as a simple extension to the branch target buffer (BTB). We use the PPG for implementing a fast pre-program counter pre-PC, that ...

Keywords: LRU mechanism, SPEC92 benchmark, Tango, base line architecture, branch target buffer, hardware resources, hardware-based data prefetching technique, instruction level parallelism, instruction prefetching, memory reference instructions, parallel processing, performance, program progress graph, simulation results, slack time, superscalar processors

7 Streamlining data cache access with fast address calculation

Todd M. Austin, Dionisios N. Pnevmatikatos, Gurindar S. Sohi

May 1995 **ACM SIGARCH Computer Architecture News , Proceedings of the 22nd annual international symposium on Computer architecture**, Volume 23 Issue 2

Full text available: pdf(1.58 MB) Additional Information: full citation, abstract, references, citings, index terms

For many programs, especially integer codes, untolerated load instruction latencies account for a significant portion of total execution time. In this paper, we present the design and evaluation of a fast address generation mechanism capable of eliminating the delays caused by effective address calculation for many loads and stores.Our approach works by predicting early in the pipeline (part of) the effective address of a memory access and using this predicted address to speculatively access the ...

8 Parallel execution of prolog programs: a survey

Gopal Gupta, Enrico Pontelli, Khayri A.M. Ali, Mats Carlsson, Manuel V. Hermenegildo

July 2001 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 23 Issue 4

Full text available: pdf(1.95 MB) Additional Information: full citation, abstract, references, citings, index terms

Since the early days of logic programming, researchers in the field realized the potential for exploitation of parallelism present in the execution of logic programs. Their high-level nature, the presence of nondeterminism, and their referential transparency, among other characteristics, make logic programs interesting candidates for obtaining speedups through parallel execution. At the same time, the fact that the typical applications of logic programming frequently involve irregular computatio ...

Keywords: Automatic parallelization, constraint programming, logic programming, parallelism, prolog

9 Experimental evaluation of on-chip microprocessor cache memories

Mark D. Hill, Alan Jay Smith

January 1984 **ACM SIGARCH Computer Architecture News , Proceedings of the 11th annual international symposium on Computer architecture**, Volume 12 Issue 3

Full text available: pdf(943.62 KB) Additional Information: full citation, abstract, references, citings, index terms

Advances in integrated circuit density are permitting the implementation on a single chip of functions and performance enhancements beyond those of a basic processors. One performance enhancement of proven value is a cache memory; placing a cache on the processor chip can reduce both mean memory access time and bus traffic. In this paper we use trace driven simulation to study design tradeoffs for small (on-chip) caches. Miss ratio and traffic ratio (bus traffic) are the metrics for cache p ...

10 System-level power optimization: techniques and tools

Luca Benini, Giovanni de Micheli

April 2000 **ACM Transactions on Design Automation of Electronic Systems (TODAES)**, Volume 5 Issue 2

Full text available: pdf(385.22 KB) Additional Information: full citation, abstract, references, citings, index terms

This tutorial surveys design methods for energy-efficient system-level design. We consider electronic sytems consisting of a hardware platform and software layers. We consider the three major constituents of hardware that consume energy, namely computation, communication, and storage units, and we review methods of reducing their energy consumption. We also study models for analyzing the energy cost of software, and methods for energy-efficient software design and compilation. This survery ...

11 Continuous program optimization: A case study

Thomas Kistler, Michael Franz

July 2003 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 25 Issue 4

Full text available: pdf(877.67 KB) Additional Information: full citation, abstract, references, index terms, review

Much of the software in everyday operation is not making optimal use of the hardware on which it actually runs. Among the reasons for this discrepancy are hardware/software mismatches, modularization overheads introduced by software engineering considerations, and the inability of systems to adapt to users' behaviors.A solution to these problems is to delay code generation until load time. This is the earliest point at which a piece of software can be fine-tuned to the actual capabilities of the ...

Keywords: Dynamic code generation, continuous program optimization, dynamic reoptimization

12 Tiny instruction caches for low power embedded systems

Ann Gordon-Ross, Susan Cotterell, Frank Vahid

November 2003 **ACM Transactions on Embedded Computing Systems (TECS)**, Volume 2 Issue 4

Full text available: pdf(887.71 KB) Additional Information: full citation, abstract, references, index terms

Instruction caches have traditionally been used to improve software performance. Recently, several tiny instruction cache designs, including filter caches and dynamic loop caches, have been proposed to instead reduce software power. We propose several new tiny instruction cache designs, including preloaded loop caches, and one-level and two-level hybrid dynamic/preloaded loop caches. We evaluate the existing and proposed designs on embedded system software benchmarks from both the Powerstone and ...

Keywords: Loop cache, architecture tuning, embedded systems., filter cache, fixed program, instruction cache, low energy, low power

13 How java programs interact with virtual machines at the microarchitectural level

Lieven Eeckhout, Andy Georges, Koen De Bosschere

October 2003 **ACM SIGPLAN Notices , Proceedings of the 18th ACM SIGPLAN conference on Object-oriented programing, systems, languages, and applications**, Volume 38 Issue 11

Full text available: pdf(348.88 KB) Additional Information: full citation, abstract, references, index terms

Java workloads are becoming increasingly prominent on various platforms ranging from embedded systems, over general-purpose computers to high-end servers. Understanding the implications of all the aspects involved when running Java workloads, is thus extremely important during the design of a system that will run such workloads. In other words, understanding the interaction between the Java application, its input and the virtual machine it runs on, is key to a succesful design. The goal of this ...

Keywords: Java workloads, performance analysis, statistical data analysis, virtual machine technology, workload characterization

14 Compiler transformations for high-performance computing

David F. Bacon, Susan L. Graham, Oliver J. Sharp

December 1994 **ACM Computing Surveys (CSUR)**, Volume 26 Issue 4

Full text available: pdf(6.32 MB) Additional Information: full citation, abstract, references, citings, index terms, review

In the last three decades a large number of compiler transformations for optimizing programs have been implemented. Most optimizations for uniprocessors reduce the number of instructions executed by the program using transformations based on the analysis of scalar quantities and data-flow techniques. In contrast, optimizations for high-performance superscalar, vector, and parallel processors maximize parallelism and memory locality with transformations that rely on tracking the properties o ...

Keywords: compilation, dependence analysis, locality, multiprocessors, optimization, parallelism, superscalar processors, vectorization

15 Fast detection of communication patterns in distributed executions

Thomas Kunz, Michiel F. H. Seuren

November 1997 **Proceedings of the 1997 conference of the Centre for Advanced Studies on Collaborative research**

Full text available: pdf(4.21 MB) Additional Information: full citation, abstract, references, index terms

Understanding distributed applications is a tedious and difficult task. Visualizations based on process-time diagrams are often used to obtain a better understanding of the execution of the application. The visualization tool we use is Poet, an event tracer developed at the University of Waterloo. However, these diagrams are often very complex and do not provide the user with the desired overview of the application. In our experience, such tools display

repeated occurrences of non-trivial commun ...

16 External memory algorithms and data structures: dealing with massive data

Jeffrey Scott Vitter
June 2001 **ACM Computing Surveys (CSUR)**, Volume 33 Issue 2

Full text available: pdf(328.46 KB) Additional Information: full citation, abstract, references, citings, index terms

Data sets in large applications are often too massive to fit completely inside the computers internal memory. The resulting input/output communication (or I/O) between fast internal memory and slower external memory (such as disks) can be a major performance bottleneck. In this article we survey the state of the art in the design and analysis of external memory (or EM) algorithms and data structures, where the goal is to exploit locality in order to reduce the I/O costs. We consider a varie ...

Keywords: B-tree, I/O, batched, block, disk, dynamic, extendible hashing, external memory, hierarchical memory, multidimensional access methods, multilevel memory, online, out-of-core, secondary storage, sorting

17 The performance advantage of applying compression to the memory system

Nihar R. Mahapatra, Jiangjiang Liu, Krishnan Sundaresan
June 2002 **ACM SIGPLAN Notices , Proceedings of the workshop on Memory system performance**, Volume 38 Issue 2 supplement

Full text available: pdf(1.34 MB) Additional Information: full citation, abstract, references

The memory system stores information comprising primarily instructions and data and secondarily address information, such as cache tag fields. It interacts with the processor by supporting related traffic (again comprising addresses, instructions, and data). Continuing exponential growth in processor performance, combined with technology, architecture, and application trends, place enormous demands on the memory system to permit this information storage and exchange at a high-enough performance ...

Keywords: Markov models, address compression, bandwidth, cache, data compression, entropy, instruction compression, latency, lossless compression, memory, register file, storage, traffic

18 Microarchitectures: Data forwarding through in-memory precomputation threads

Wessam Hassanein, José Fortes, Rudolf Eigenmann
June 2004 **Proceedings of the 18th annual international conference on Supercomputing**

Full text available: pdf(148.05 KB) Additional Information: full citation, abstract, references, index terms

In modern architectures, memory access latency is an increasingly performance-limiting factor. To reduce this latency, we propose concepts and implementation of a new technique that uses an in-memory processor to precompute future, critical load addresses and forward the computed values to the main processor. The acronym for this technique is IMPT for In-Memory Precomputation-based forwarding Threads. IMPT combines the advantages of precomputation-based techniques with the low memory access late ...

Keywords: forwarding, precomputation, prefetching, processing-in-memory

19 Low power memory system: Low-power data memory communication for application-specific embedded processors

Peter Petrov, Alex Orailoglu

October 2002 **Proceedings of the 15th international symposium on System Synthesis**

Full text available: pdf(104.64 KB) Additional Information: full citation, abstract, references, index terms

We propose a novel customization methodology for power reduction on the communication link between an embedded processor and its data memory. We target the address bus and show how by utilizing application information about the memory references in the data intensive program loops, a power efficient address communication protocol can be established between the processor core and the data memory. The data memory controller thus generates the addresses for the various data streams with minimal run ...

20 Dynamic speculation and synchronization of data dependences

Andreas Moshovos, Scott E. Breach, T. N. Vijaykumar, Gurindar S. Sohi

May 1997 **ACM SIGARCH Computer Architecture News , Proceedings of the 24th annual international symposium on Computer architecture**, Volume 25 Issue 2

Full text available: pdf(2.51 MB) Additional Information: full citation, abstract, references, citings, index terms

Data dependence speculation is used in instruction-level parallel (ILP) processors to allow early execution of an instruction before a logically preceding instruction on which it may be data dependent. If the instruction is independent, data dependence speculation succeeds; if not, it fails, and the two instructions must be synchronized. The modern dynamically scheduled processors that use data dependence speculation do so blindly (i.e., every load instruction with unresolved dependences is spec ...